Computational Visual Media
DOI 10.1007/s41095-016-0056-2 Vol. 2, No. 3, September 2016, 267–276

Research Article

Removing mixed noise in low rank textures by convex optimization

Xiao Liang[1] (✉)

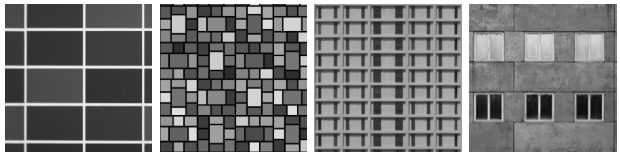
Abstract This paper introduces a new low rank texture image denoising algorithm, which can restore low rank texture contaminated by both Gaussian and salt-and-pepper noise. The algorithm formulates texture image denoising in terms of solving a low rank matrix optimization problem. Simply assuming low rank is insufficient to describe the properties of natural images, causing high noise amplitudes which lead to unsatisfactory denoising results or serious loss of image details. Thus, in addition to the low rank assumption, the continuity of natural images is also assumed by the algorithm, by adding a total variation regularizer to the optimization objective function. We further give an effective algorithm to solve this optimization problem. By combining the low rank and continuity assumptions, the proposed algorithm overcomes the deficiencies of using either the low rank assumption or total variation regularization alone. Experiments show that our algorithm can effectively remove mixed noise in low rank texture images, and is better than existing algorithms in both its subjective visual effects and in terms of quantitative objective measures.

Keywords image denoising; low rank texture; total variation; convex optimization; augmented Lagrangian method

1 Introduction

Image denoising is an extensively studied problem in the image processing community and continues to attract researchers who aim to perform better restoration in the presence of noise. During the past few decades, many intelligent methods have been proposed to improve single-image denoising performance. From pixel level filtering methods, such as Gaussian filtering [1], bilateral filtering, and total variation regularization [2], to patch based filtering methods, such as non-local means [3], block-matching 3D filtering (BM3D) [4], and sparse representation [5], single-image based denoising performance has been greatly improved, with image details well recovered when the image is slightly noisy. As such filtering methods are widely used in computer vision, work has considered how to speed them up, e.g., for the bilateral filter [6] and weighted median [7]. Comprehensive overviews of image denoising methods can be found in Refs. [8, 9].

Most of the approaches mentioned above consider the input image as an ordinary signal, taking the image as a vector or a set of patches. They do explicitly utilize internal structural information in the image. However, a typical 3D scene of an artificial environment is rich in regular structures. For instance, in an urban environment, the scene is typically filled with man-made objects that have parallel edges, right-angled corners, regular shapes, symmetric structures, and repeated patterns. Developing algorithms targeted to such images is necessary. In this paper, we mainly study images or textures with low rank structure (see Fig. 1 for examples). A rigorous definition [10] of low rank texture is given in Section 2.



(a) (b) (c) (d)

Fig. 1 Representative examples of low rank textures. These provide initial images for Figs. 3–5.

1 Institute for Advanced Study, Tsinghua University, Beijing 100084, China. E-mail: liangx04@mails.tsinghua.edu.cn (✉).

Manuscript received: 2016-04-22; accepted: 2016-05-25

Recently, low rank matrix denoising algorithms have been widely studied [11–15]. The traditional, image denoising algorithms based on low rank matrix recovery only have the low rank constraint. When Gaussian noise becomes too large, these algorithms produce unsatisfactory denoising results, with serious loss of image details. The reason is partially because that most algorithms use the nuclear norm of the matrix to approximate its rank, in order to get a soluble convex object function, which brings the following problem: as the amplitude of the Gaussian noise increases, the energy of the matrix is compressed more and more seriously. In order to solve the problem of texture image denoising in a mixed noise model, we propose a new algorithm call LRTD (*low rank texture denoising*) in this paper.

LRTD formulates texture image denoising in terms of solving a low rank matrix optimization problem. Because the low rank assumption is not sufficient by itself to describe the properties of natural images [16], high noise amplitudes will lead to unsatisfactory denoising results with serious loss of image details. Thus, in addition to the low rank assumption, we also assume image continuity in the algorithm, by adding a total variation regularizer to the optimization objective function. An effective algorithm to solve this optimization problem is also given in the paper. By combining the low rank assumption and the continuity assumption, the proposed LRTD algorithm can overcome the deficiencies of assuming low rank assumption or using total variation regularization alone. The algorithm can effectively remove mixed noise in low rank texture images, and is better than existing algorithms in terms of both subjective visual effects and objective quantitative measurements.

2 Definition of low rank textures

In this paper, we consider a 2D texture as a function $I^0(x, y)$, defined on $\mathbb{R}^2$. We say that I^0 is a *low rank texture* if the family of one-dimensional functions $\{I^0(x, y_0) \mid y_0 \in \mathbb{R}\}$ span a finite low-dimensional linear subspace:

$$r \doteq \dim\left(\operatorname{span}\{I^0(x, y_0) \mid y_0 \in \mathbb{R}\}\right) \leqslant k$$

for some small positive integer k. If r is finite, then we refer to I^0 as a rank-r texture. Figure 1 shows some ideal low rank textures. To a large extent, the notion of low rank texture unifies many conventional local features. Using this definition, it is easy to see that *images of regular symmetric patterns always lead to low rank textures*. Thus, the notion of low rank texture encompasses a much broader range of "features" or regions than corners and edges.

3 Problem formulation

Before we statistically analyze image denoising, we first define our image formation model:

$$I = L + N$$

where $I \in \mathbb{R}^{m\times n}$ is the observed noisy image matrix, $L \in \mathbb{R}^{m\times n}$ is the noise free low rank texture image, and $N \in \mathbb{R}^{m\times n}$ is the noise matrix. Here we assume the noise model is a mixture of noise: $N = E + Z$, where $Z \in \mathbb{R}^{m\times n}$ is additive zero-mean independent identically distributed (i.i.d.) Gaussian noise with variance σ^2; $E \in \mathbb{R}^{m\times n}$ is random impulsive noise (sparse but large), which is chosen uniformly at random, and the non-zero entries of E are i.i.d. uniformly in the interval $[0, 255]$. The denoising problem can be modeled as the following optimization problem:

$$\begin{aligned} &\min_L \operatorname{rank}(L) + \lambda\|E\|_0 \\ &\text{such that } L + E + Z = I, \\ &\qquad Z \in B : \{Z \in \mathbb{R}^{m\times n} \mid \|Z\|_{\mathrm{F}} \leqslant \delta\} \end{aligned} \tag{1}$$

where $\|E\|_0$ denotes the number of non-zero entries in E, $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm, $\delta > 0$ is a Gaussian noise intensity parameter, and λ is a weighting parameter which trades off the rank and sparsity of the recovered image. In the above problem, both the rank function and the l_0 norm can be replaced by convex surrogates [17]: the matrix nuclear norm[1] $\|L\|_*$ for $\operatorname{rank}(L)$ and the l_1 norm[2] $\|E\|_1$ for $\|E\|_0$, respectively. Thus, we end up with the following optimization problem:

$$\begin{aligned} &\min_L \|L\|_* + \lambda\|E\|_1 \\ &\text{such that } L + E + Z = I, \\ &\qquad Z \in B : \{Z \in \mathbb{R}^{m\times n} \mid \|Z\|_{\mathrm{F}} \leqslant \delta\} \end{aligned} \tag{2}$$

Formulation (2) utilizes the low rank nature of the image and the sparsity of the impulsive noise E. But as noted in Ref. [16], while being low rank is a necessary condition for most regular, structured images, it is certainly not sufficient. We need other priors to model additional structures in

[1] The nuclear norm of a matrix is the sum of all its singular values.
[2] The l_1 norm is the sum of absolute values.

the natural image. Moreover, because the nuclear norm is an approximation to the rank of a matrix, when the noise amplitude is large, formulation (2) leads to over-compression of the nuclear norm [18], causing the total energy of the denoised image to significantly decrease: the picture becomes darker; see for example the fourth column of Fig. 5. Thus, to take into account the piecewise smooth continuity of a natural image, we add a total variation regularizer to the optimization problem:

$$\begin{aligned}&\min_L \|L\|_* + \lambda\|E\|_1 + \alpha\|L\|_{\mathrm{TV}}\\ &\text{such that } L+E+Z=I,\\ &\qquad Z\in B:\{Z\in\mathbb{R}^{m\times n}\mid \|Z\|_{\mathrm{F}}\leqslant\delta\}\end{aligned}\tag{3}$$

where $\|L\|_{\mathrm{TV}} = \|D_xL\|_1 + \|D_yL\|_1$ is the total variation regularizer, in which D_x and D_y are first order forward finite-difference operators in horizontal and vertical directions respectively. Their definitions are

$$\begin{aligned}D_xL &= \mathrm{vec}(L(x+1,y)-L(x,y))\\ D_yL &= \mathrm{vec}(L(x,y+1)-L(x,y))\\ D_x^{\mathrm{T}}L &= \mathrm{vec}(L(x-1,y)-L(x,y))\\ D_y^{\mathrm{T}}L &= \mathrm{vec}(L(x,y-1)-L(x,y))\end{aligned}$$

with periodic boundary conditions; $\mathrm{vec}(\cdot)$ represents the vectorization operator.

4 Denoising by convex optimization

To solve the convex optimization problem in Eq. (3), we use the *alternating direction method* (ADM) [19], as it has been proven to be one of the fastest algorithms for solving various low rank matrix completion and recovery problems. To be able to adopt the ADM method to our problem, we need to make our objective function separable. Thus we introduce three auxiliary variables Cx, Cy, and W, which turns the optimization problem into the following:

$$\begin{aligned}&\min_L \|L\|_* + \lambda\|E\|_1 + \alpha\|Cx\|_1 + \alpha\|Cy\|_1\\ &\text{such that } Cx=D_xL, Cy=D_yL, L+E+Z=I,\\ &\qquad L=W, Z\in B:\{Z\in\mathbb{R}^{m\times n}\mid \|Z\|_{\mathrm{F}}\leqslant\delta\}\end{aligned}\tag{4}$$

In formulation (4), the augmented Lagrangian function is defined as

$$\begin{aligned}&L_\mu(L,W,E,Z,Cx,Cy,Y_1,Y_2,Y_3,Y_4) =\\ &\quad \|L\|_* + \lambda\|E\|_1 + \alpha\|Cx\|_1 + \alpha\|Cy\|_1\\ &\quad +\langle Y_1, Cx-D_xL\rangle + \langle Y_2, Cy-D_yL\rangle\\ &\quad +\langle Y_3, L+E+Z-I\rangle + \langle Y_4, L-W\rangle\\ &\quad +\frac{\mu}{2}\|Cx-D_xL\|_{\mathrm{F}}^2 + \frac{\mu}{2}\|Cy-D_yL\|_{\mathrm{F}}^2\\ &\quad +\frac{\mu}{2}\|L+E+Z-I\|_{\mathrm{F}}^2 + \frac{\mu}{2}\|L-W\|_{\mathrm{F}}^2\end{aligned}\tag{5}$$

where L, W, E, Z, Cx, Cy are the unknown variables, Y_1, Y_2, Y_3, Y_4 are Lagrange multipliers, and $\mu > 0$ is a penalty parameter; $\langle\cdot,\cdot\rangle$ indicates inner product. The resulting classic ADM iteration scheme for our problem is given by

$$\begin{aligned}L_{k+1} &= \arg\min_L L_{\mu_k}(L, W_k, E_k, Z_k, Cx_k, Cy_k,\\ &\qquad Y_{1,k}, Y_{2,k}, Y_{3,k}, Y_{4,k})\end{aligned}\tag{6}$$

$$\begin{aligned}Cx_{k+1} &= \arg\min_{Cx} L_{\mu_k}(L_{k+1}, W_k, E_k, Z_k, Cx, Cy_k,\\ &\qquad Y_{1,k}, Y_{2,k}, Y_{3,k}, Y_{4,k})\end{aligned}\tag{7}$$

$$\begin{aligned}Cy_{k+1} &= \arg\min_{Cy} L_{\mu_k}(L_{k+1}, W_k, E_k, Z_k, Cx_{k+1},\\ &\qquad Cy, Y_{1,k}, Y_{2,k}, Y_{3,k}, Y_{4,k})\end{aligned}\tag{8}$$

$$\begin{aligned}E_{k+1} &= \arg\min_E L_{\mu_k}(L_{k+1}, W_k, E, Z_k, Cx_{k+1},\\ &\qquad Cy_{k+1}, Y_{1,k}, Y_{2,k}, Y_{3,k}, Y_{4,k})\end{aligned}\tag{9}$$

$$\begin{aligned}W_{k+1} &= \arg\min_W L_{\mu_k}(L_{k+1}, W, E_{k+1}, Z_k, Cx_{k+1},\\ &\qquad Cy_{k+1}, Y_{1,k}, Y_{2,k}, Y_{3,k}, Y_{4,k})\end{aligned}\tag{10}$$

$$\begin{aligned}Z_{k+1} &= \arg\min_Z L_{\mu_k}(L_{k+1}, W_{k+1}, E_{k+1}, Z,\\ &\qquad Cx_{k+1}, Cy_{k+1}, Y_{1,k}, Y_{2,k}, Y_{3,k}, Y_{4,k})\end{aligned}\tag{11}$$

$$\begin{aligned}Y_{1,k+1} &= Y_{1,k} + \mu_k(Cx_{k+1} - D_xL_{k+1})\\ Y_{2,k+1} &= Y_{2,k} + \mu_k(Cy_{k+1} - D_yL_{k+1})\\ Y_{3,k+1} &= Y_{3,k} + \mu_k(L_{k+1} + E_{k+1} + Z_{k+1} - I)\\ Y_{4,k+1} &= Y_{4,k} + \mu_k(L_{k+1} - W_{k+1})\\ \mu_{k+1} &= \rho\mu_k\end{aligned}$$

where $\rho > 1$ is a constant. We now focus on efficiently solving the first six steps of the above iterative scheme.

1) Solving Eq. (6)

$$\begin{aligned}L_{k+1} &= \arg\min_L \|L\|_* + \frac{\mu_k}{2}\|L + E_k + Z_k - I\\ &\qquad + Y_{3,k}/\mu_k\|_{\mathrm{F}}^2 + \frac{\mu_k}{2}\|L - W_k + Y_{4,k}/\mu_k\|_{\mathrm{F}}^2\\ &= \boldsymbol{S}_{(\mu_k)^{-1}}\left(\frac{I - E_k - Z_k - \frac{1}{\mu_k}Y_{3,k} + W_k - \frac{1}{\mu_k}Y_{4,k}}{2}\right)\end{aligned}$$

where $\boldsymbol{S}_{(\mu_k)^{-1}}(\cdot)$ is the singular value shrinkage operator:

$$\boldsymbol{S}_\varepsilon(X) = U\boldsymbol{T}_\varepsilon(\Sigma)V^{\mathrm{T}}$$

in which $U\Sigma V^{\mathrm{T}}$ is the SVD (singular value decomposition) of X, and $\boldsymbol{T}[\cdot]$ represents the soft-thresholding operator defined for scalars as follows:

$$\boldsymbol{T}_\varepsilon[x] = \mathrm{sign}(x)\,(|x| - \varepsilon)$$

for $\varepsilon \geqslant 0$; it is extended to vectors and matrices by applying it elementwise.

2) Solving Eqs. (7)–(9)

Each of these three variables has closed form solutions, as follows:

$$Cx_{k+1} = \arg\min_{Cx} \alpha\|Cx\|_1 + \frac{\mu_k}{2}\|Cx - D_xW_k + Y_{1,k}/\mu_k\|_{\mathrm{F}}^2 = \boldsymbol{T}_{\alpha/\mu_k}(D_xW_k - Y_{1,k}/\mu_k)$$

$$Cy_{k+1} = \arg\min_{Cy} \alpha\|Cy\|_1 + \frac{\mu_k}{2}\|Cy - D_yW_k + Y_{2,k}/\mu_k\|_{\mathrm{F}}^2 = \boldsymbol{T}_{\alpha/\mu_k}(D_yW_k - Y_{2,k}/\mu_k)$$

$$E_{k+1} = \arg\min_{E} \lambda\|E\|_1 + \frac{\mu_k}{2}\|L_{k+1} + E + Z_k - I + Y_{3,k}/\mu_k\|_{\mathrm{F}}^2 = \boldsymbol{T}_{\lambda/\mu_k}(I - L_{k+1} - Z_k - Y_{3,k}/\mu_k)$$

3) Solving Eq. (10)

$$W_{k+1} = \arg\min_{W} \frac{\mu_k}{2}\|W - L_{k+1} + Y_{4,k}/\mu_k\|_{\mathrm{F}}^2 + \frac{\mu_k}{2}\|Cx_{k+1} - D_xW + Y_{1,k}/\mu_k\|_{\mathrm{F}}^2 + \frac{\mu_k}{2}\|Cy_{k+1} - D_yW + Y_{2,k}/\mu_k\|_{\mathrm{F}}^2$$

Here W also has a closed form solution:

$$(\mathrm{Id} + D_x^{\mathrm{T}}D_x + D_y^{\mathrm{T}}D_y)W_{k+1} = D_x^{\mathrm{T}}(Cx_{k+1} + Y_{1,k}/\mu_k) + D_y^{\mathrm{T}}(Cy_{k+1} + Y_{2,k}/\mu_k) + L - Y_{4,k}/\mu_k$$

where Id is the identity matrix. Then we use Fourier transform to solve W [2]:

$$W_{k+1} = \mathcal{F}^{-1}\left[\frac{\mathcal{F}[D_x^{\mathrm{T}}(Cx_{k+1}+Y_{1,k}/\mu_k)+D_y^{\mathrm{T}}(Cy_{k+1}+Y_{2,k}/\mu_k)+L-Y_{4,k}/\mu_k]}{\mathcal{F}[\mathrm{Id}+D_x^{\mathrm{T}}D_x+D_y^{\mathrm{T}}D_y]}\right] \tag{12}$$

where $\mathcal{F}$ denotes the 2D Fourier transform operator. The denominator on the right hand side of Eq. (12) is independent of the iteration number k, and so can be precalculated outside the main loop. Therefore, the complexity of solving Eq. (12) is the complexity of one 2D Fourier transform and one inverse 2D Fourier transform.

4) Solving Eq. (11)

Following Ref. [13], we write

$$Z_{k+1} = \arg\min_{Z} \frac{\mu_k}{2}\|L_{k+1} + E_{k+1} + Z - I + Y_{3,k}/\mu_k\|_{\mathrm{F}}^2 = \frac{\min\{\|N\|_{\mathrm{F}}, \delta\}}{\|N\|_{\mathrm{F}}}N$$

$$N = I - L_{k+1} - E_{k+1} - Y_{3,k}/\mu_k$$

Algorithm 1 gives pseudocode of the overall LRTD algorithm.

5 Results

In this section we compare our LRTD algorithm with existing approaches. All experiments were performed using MATLAB on a laptop with a 2.30 GHz processor and 8 GB of RAM.

The parameter setting used for our algorithm were $\lambda = 2.5/\sqrt{\max(m,n)}$, $\alpha = 0.1/\sqrt{\max(m,n)}$, $\mu_0 = 1.25/\|I\|_2$, $\rho = \max(1.4 - \sigma/600, 1.2)$. As clarified in Ref. [20], if the Gaussian noise is white noise with standard deviation σ, then the parameter δ in Eq. (1) satisfies $\delta^2 \leqslant (d + \sqrt{8d})\sigma^2$ with high probability, in which $d = m \times n$ is the number of pixels in the input image I. Therefore, we set $\delta = \sqrt{(d + \sqrt{8d})}\sigma$.

We select a set of parameters with the best overall performance for λ and α in our algorithm; LRTD is not sensitive to parameter μ_0. $\rho = \max(1.4 -$

Algorithm 1: ADM algorithm for solving problem (4)

Input: Input image $I \in \mathbb{R}^{m\times n}$, parameters $\lambda > 0$, $\alpha > 0$.

Initialize: $k = 0, L_0 = I, E_0 = 0, Z_0 = 0, W_0 = 0, Cx = D_xI, Cy = D_yI, Y_{1,0} = 0, Y_{2,0} = 0, Y_{3,0} = 0, Y_{4,0} = 0$, $\mu_0 > 0, \rho > 1$.

WHILE $\|L_{k+1} - L_k\|_2/\|L_k\|_2 \geqslant$ tolerance **DO**

$L_{k+1} = \boldsymbol{S}_{(\mu_k)^{-1}}\left(I - E_k - Z_k - \frac{1}{\mu_k}Y_{3,k} + W_k - \frac{1}{\mu_k}Y_{4,k}/2\right)$;

$Cx_{k+1} = \boldsymbol{T}_{\alpha/\mu_k}(D_xW_k - Y_{1,k}/\mu_k)$;

$Cy_{k+1} = \boldsymbol{T}_{\alpha/\mu_k}(D_yW_k - Y_{1,k}/\mu_k)$;

$E_{k+1} = \boldsymbol{T}_{\lambda/\mu_k}(I - L_{k+1} - Z_k - Y_{3,k}/\mu_k)$;

$W_{k+1} = \mathcal{F}^{-1}\left[\frac{\mathcal{F}[D_x^{\mathrm{T}}(Cx_{k+1}+Y_{1,k}/\mu_k)+D_y^{\mathrm{T}}(Cy_{k+1}+Y_{2,k}/\mu_k)+L-Y_{4,k}/\mu_k]}{\mathcal{F}[\mathrm{Id}+D_x^{\mathrm{T}}D_x+D_y^{\mathrm{T}}D_y]}\right]$;

$Z_{k+1} = \frac{\min\{\|N\|_{\mathrm{F}},\delta\}}{\|N\|_{\mathrm{F}}}N$;

$Y_{1,k+1} = Y_{1,k} + \mu_k \cdot (Cx_{k+1} - D_xL_{k+1})$;

$Y_{2,k+1} = Y_{2,k} + \mu_k \cdot (Cy_{k+1} - D_yL_{k+1})$;

$Y_{3,k+1} = Y_{3,k} + \mu_k \cdot (L_{k+1} + E_{k+1} + Z_{k+1} - I)$;

$Y_{4,k+1} = Y_{4,k} + \mu_k \cdot (L_{k+1} - W_{k+1})$;

$\mu_{k+1} = \rho\mu_k$;

END WHILE

Output: Solution (L, E, W, Z, Cx, Cy) to problem (4).

$\sigma/600, 1.2)$ is related to noise intensity σ. The greater the σ, the smaller the ρ.

5.1 Comparison with other mixed noise removal methods

In this paper we consider image denoising problems with mixed noise, in which the image is contaminated by both Gaussian white noise and salt-and-pepper noise. Some well known denoising methods such as BM3D work very well to restore images contaminated by pure Gaussian noise, but are completely unable to deal with salt-and-pepper noise [21]. Thus, we only compare our LRTD method with three other noise removal methods [13, 22, 23] specifically designed for mixed noise.

Following their papers, the parameter settings used were: for Ref. [22], $\beta_2 = 0.00002$, $tol = 10^{-4}$, $\eta = 1$; for Ref. [23], outPer $= sr$, blocksize $= [8, 8]$, stepsize $= [2, 2]$; for Ref. [13], $\eta = 1.3$, $\beta = 0.13\beta_0$. Besides visual comparison of the results, peak signal to noise ratio (PSNR) was measured to quantitatively evaluate the quality of the restoration results. Given an image $L^* \in [0, 255]^{m\times n}$, the PSNR of its estimated L is defined as

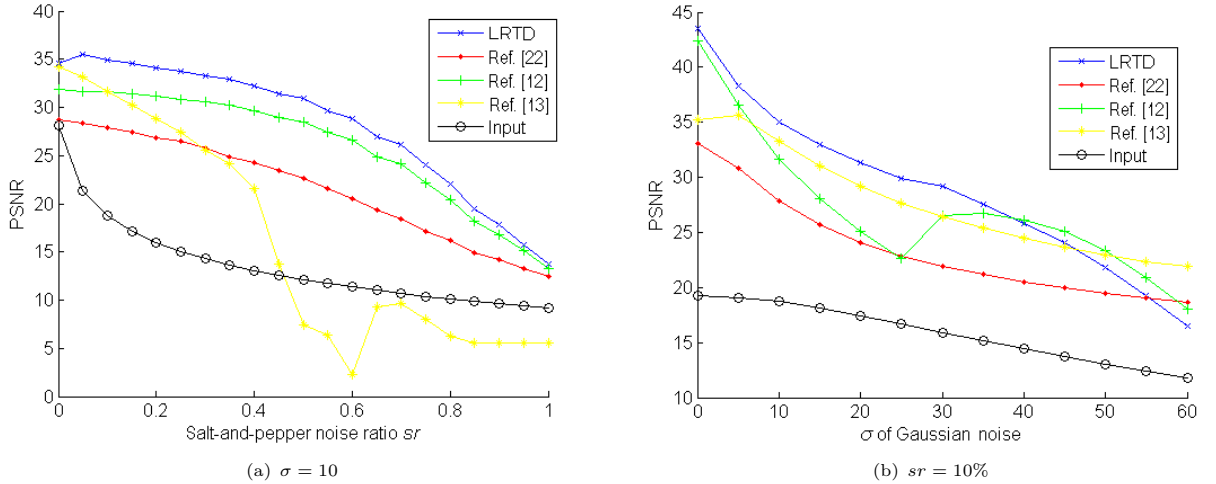
$$\mathrm{PSNR}(L^*, L) = 10\log_{10}\frac{255^2}{\frac{1}{m\times n}\sum_{i,j}(L^*_{i,j} - L_{i,j})^2}.$$

A quantitative comparison is shown in Fig. 2. Here, we used the image in Fig. 1(c) for testing. The percentage of salt-and-pepper noise pixels in the image is denoted by sr (salt-and-pepper noise ratio), while the standard deviation of the white Gaussian noise is denoted by σ. Figure 3(a) shows how PSNR varies for the denoised images as the salt-and-pepper noise ratio sr varies from 0 to 100% with fixed Gaussian noise $\sigma = 10$; Fig. 3(b) shows how PSNR of denoised images varies as the standard deviation of the Guassian noise σ varies from 0 to 60 with a fixed salt-and-pepper noise percentage $sr = 10\%$.

Further quantitative results for the four algorithms were obtained using input images generated by adding salt-and-pepper noise with different levels $(sr = 15\%, 30\%, 45\%)$ mixed with Gaussian noise with different levels $(\sigma = 5, 15, 30, 60)$ to the textures shown in Fig. 1. The PSNR values of the restoration results of these methods are summarized in Table 1. Figures 3–5 give qualitative comparisons; the original images in these experiment are shown in Fig. 1. We can see from these results that LRTD works better on low rank texture images than previous algorithms.

From Fig. 2 and Table 1 we can see that LRTD's ability to process salt-and-pepper noise is very good. In low Gaussian noise environments $(\sigma < 10)$, as the percentage of salt-and-pepper noise increases from 0% to 60%, the PSNRs of our image restoration results do not decrease significantly, and are always more than 30 dB. However, the LRTD algorithm is more sensitive to Gaussian noise. As Gaussian noise increases to about $\sigma = 60$, results shown in Fig. 5 display the excessive compression issue caused by use of convex surrogates for the rank function. Although the structure of the restored image is still quite good, due to the compression of the overall energy of the input image during the optimization process, the resulting PSNR decreases significantly.

The results in this subsection demonstrate that our LRTD denoising method can effectively remove



(a) $\sigma = 10$ (b) $sr = 10\%$

Fig. 2 Variation of PSNR with varying amounts of salt-and-pepper noise and Gaussian noise. (a) $\sigma = 10$, sr varying from 0 to 100%; (b) $sr = 10\%$, σ varying from 0 to 60.

Table 1 Comparison with other mixed noise removal methods, showing PSNR values for varying amounts of salt-and-pepper and Gaussian noise

σ	Image	$sr = 15\%$				$sr = 30\%$				$sr = 45\%$			
		Ref. [22]	Ref. [23]	Ref. [13]	LRTD	Ref. [22]	Ref. [23]	Ref. [13]	LRTD	Ref. [22]	Ref. [23]	Ref. [13]	LRTD
5	Image1	32.17	34.42	36.73	**39.47**	29.16	28.52	30.40	**38.38**	25.87	14.85	24.45	**36.14**
	Image2	29.13	27.13	36.94	**40.19**	25.41	22.39	32.75	**37.71**	22.06	11.42	24.90	**33.06**
	Image3	29.78	34.20	36.21	**37.85**	27.03	28.43	34.96	**36.48**	24.43	20.37	32.60	**34.61**
	Image4	29.30	31.89	32.75	**33.61**	27.44	28.04	31.06	**32.10**	25.03	19.80	28.53	**30.06**
15	Image1	26.53	32.96	26.86	**33.63**	25.60	26.41	25.21	**32.11**	23.47	16.66	22.24	**30.01**
	Image2	25.59	28.61	26.61	**32.22**	23.43	23.16	24.45	**29.59**	20.70	11.42	21.06	**26.28**
	Image3	25.35	29.99	27.99	**32.20**	24.11	25.75	27.43	**30.98**	22.45	19.76	25.31	**29.12**
	Image4	25.39	30.12	27.07	**29.96**	24.53	26.47	25.90	**28.57**	23.22	17.36	23.90	**26.72**
30	Image1	22.70	28.80	27.51	**30.50**	22.10	23.77	23.69	**27.28**	20.30	18.21	18.19	**24.71**
	Image2	22.11	26.31	25.13	**27.63**	20.75	21.55	21.36	**24.47**	18.73	14.12	16.88	**21.40**
	Image3	21.74	25.65	20.86	**28.07**	21.01	22.69	21.08	**26.53**	19.94	19.19	19.19	**24.19**
	Image4	22.19	26.86	20.77	**26.70**	21.68	24.21	20.24	**25.17**	20.73	20.96	19.23	**23.22**
60	Image1	19.32	24.12	20.50	**24.20**	18.36	20.58	19.39	**21.09**	16.87	16.72	18.27	**18.37**
	Image2	18.66	**21.85**	18.25	20.80	17.49	**18.97**	17.35	18.40	15.97	14.73	15.77	**16.05**
	Image3	18.61	21.24	20.55	**23.32**	18.02	19.43	20.21	**21.44**	17.06	17.18	18.84	**19.44**
	Image4	19.11	**23.41**	18.67	22.41	18.72	**21.62**	19.14	21.46	17.91	19.18	18.50	**19.85**

mixed noise in low rank texture images, and works better than other existing algorithms. Addition of the TV regularizer to the optimization objective function has a good effect on avoiding the problem of excessive compression. Our LRTD shows significant improvement compared to the simple low rank optimization algorithm [13].

5.2 Computation time

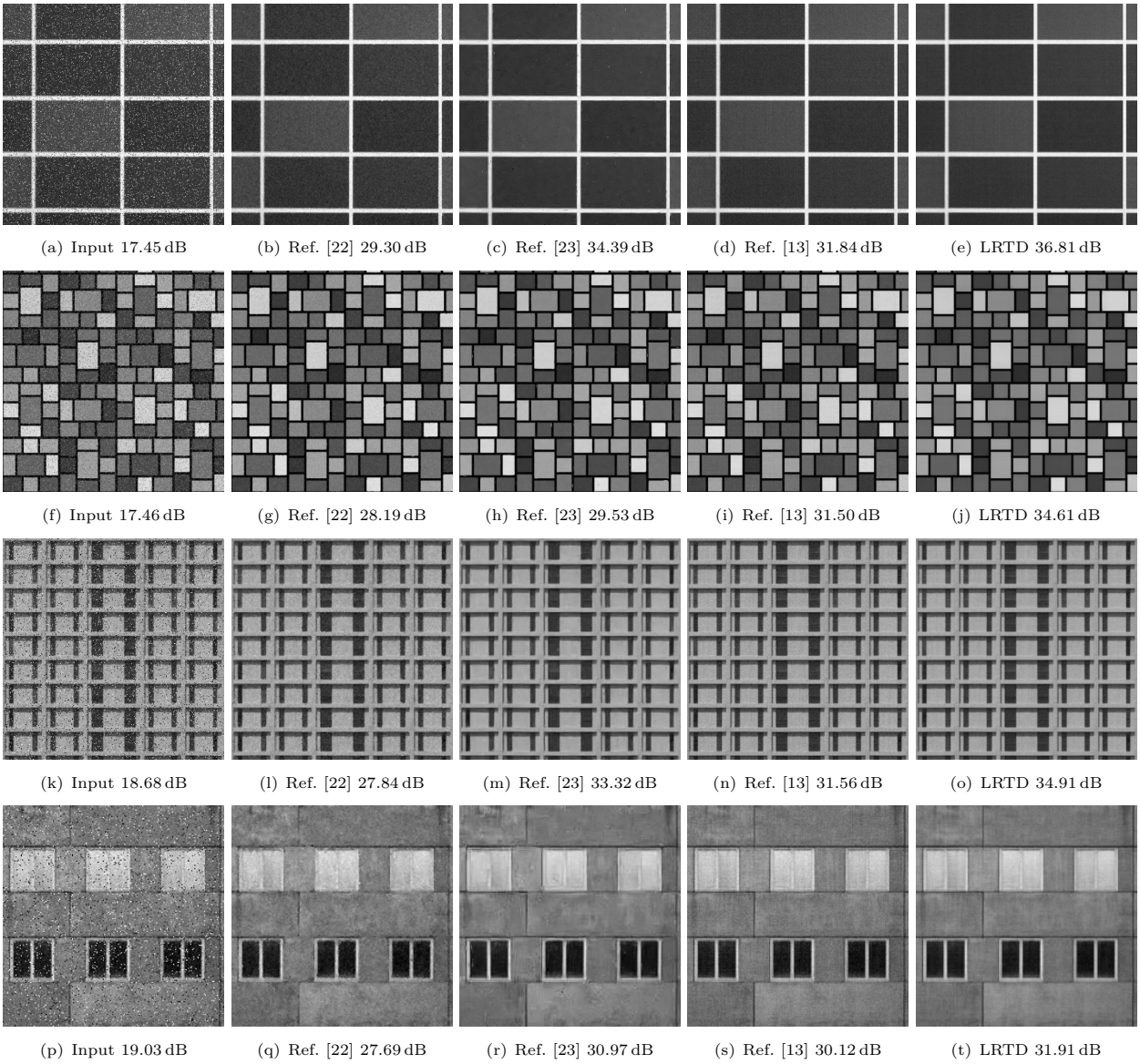
We performed a further experiment to test the convergence performance of LRTD. Let $I = L^* + E^* + Z^*$ be the noisy data matrix, where L^* and E^* are the low rank and sparse components to be recovered. We generated a rank 2 checkerboard image as $L^* \in \mathbb{R}^{320\times320}$. The support Ω of the impulsive noise E^* (sparse but large) was chosen uniformly at random, and the non-zero entries of E^* were i.i.d. uniform in the interval $[-500, 500]$.

As our aim was to test running speed and convergence of LRTD, we set $\sigma = 0$; this differs from the mixed noise used in the earlier tests as there is no Gaussian noise. In this case, the algorithm can converge to the ground truth noiseless matrix L^*.

Here we compare our method with another low rank matrix recovery algorithm ASALM [13]. The objective function of ASALM is given in Eq. (2): the only difference between ASALM with our LRTD is that ASALM lacks the TV regularizer. Following their paper, we set the parameters for ASALM to $\eta = 1.3$, $\beta = 0.13\beta_0$. Table 2 shows that LRTD is slower than ASALM. However, LRTD performs better in denoising. We believe the trade-off to be acceptable.

Table 2 Computation time and number of iterations

sr	Algorithm	100×100			200×200			400×400		
		Time (s)	Iteration	$\frac{\|L-L^*\|_F}{\|L\|_F}$	Time (s)	Iteration	$\frac{\|L-L^*\|_F}{\|L\|_F}$	Time (s)	Iteration	$\frac{\|L-L^*\|_F}{\|L\|_F}$
0.05	LRTD	0.674	54	3.081e-07	2.090	55	2.663e-07	9.104	55	2.837e-07
	ASALM [13]	0.193	36	2.285e-07	0.791	37	2.941e-05	4.623	40	7.067e-06
0.1	LRTD	0.627	57	2.000e-07	1.857	57	2.369e-07	9.152	57	2.424e-07
	ASALM [13]	0.217	38	1.154e-06	0.853	40	2.068e-07	5.053	43	5.182e-07
0.15	LRTD	0.589	61	6.41e-08	1.831	59	1.647e-07	9.028	58	2.423e-07
	ASALM [13]	0.198	39	4.179e-08	1.042	44	3.22e-05	5.276	46	1.274e-07
0.2	LRTD	0.601	61	3.607e-05	1.894	61	2.078e-07	9.298	60	1.722e-07
	ASALM [13]	0.239	42	4.8336e-07	0.939	43	5.931e-07	5.505	40	2.278e-07



(a) Input 17.45 dB (b) Ref. [22] 29.30 dB (c) Ref. [23] 34.39 dB (d) Ref. [13] 31.84 dB (e) LRTD 36.81 dB

(f) Input 17.46 dB (g) Ref. [22] 28.19 dB (h) Ref. [23] 29.53 dB (i) Ref. [13] 31.50 dB (j) LRTD 34.61 dB

(k) Input 18.68 dB (l) Ref. [22] 27.84 dB (m) Ref. [23] 33.32 dB (n) Ref. [13] 31.56 dB (o) LRTD 34.91 dB

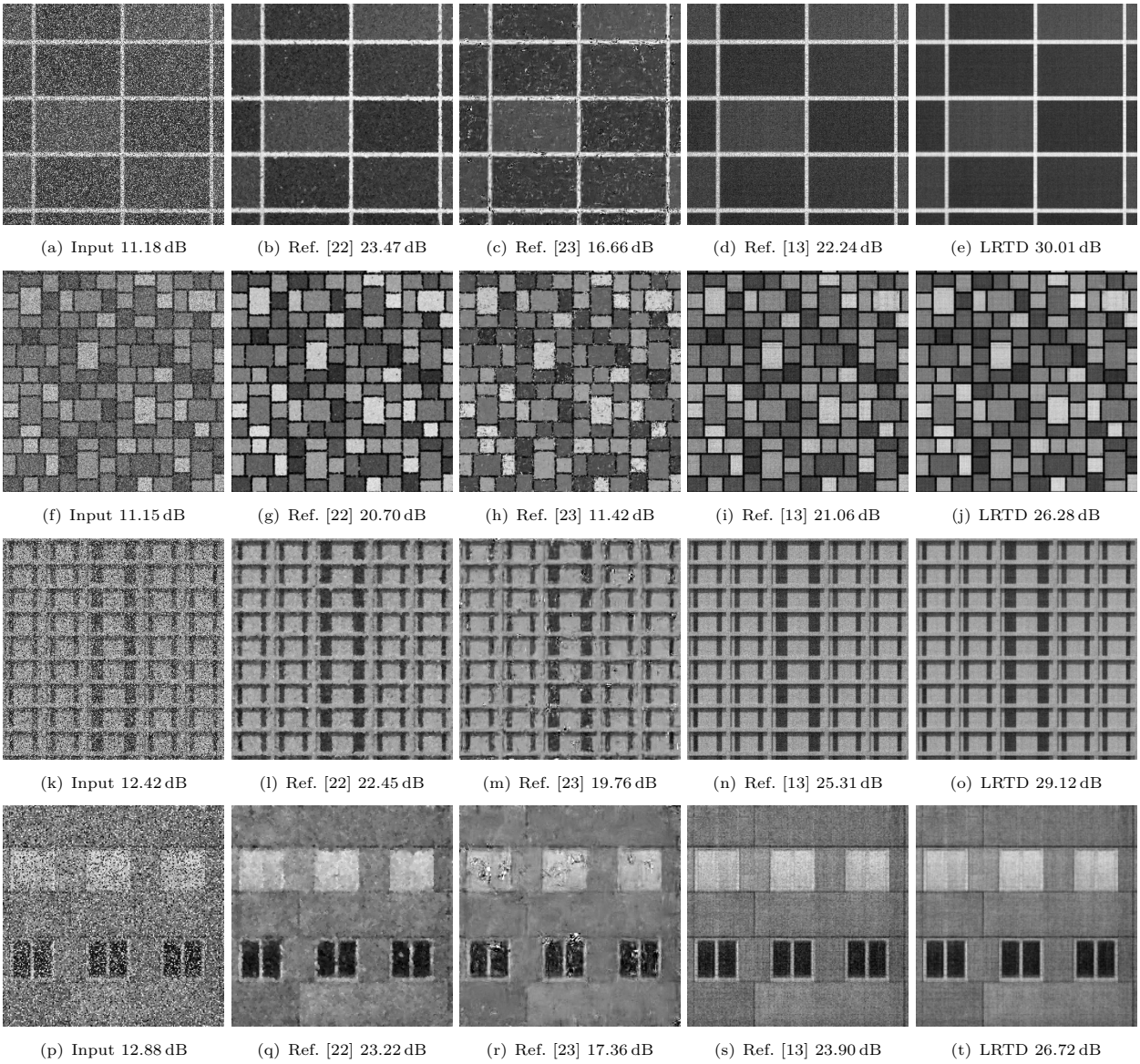
(p) Input 19.03 dB (q) Ref. [22] 27.69 dB (r) Ref. [23] 30.97 dB (s) Ref. [13] 30.12 dB (t) LRTD 31.91 dB

Fig. 3 Qualitative comparison of results of various methods (1). Left to right: input noisy image with $\sigma = 10, sr = 10\%$, denoising results of Ref. [22], KALS [23], ASALM [13], and our denoising result.

6 Conclusions and discussion

This paper introduced a new low rank texture image denoising algorithm, which can restore low rank texture contaminated by both Gaussian and salt-and-pepper noise. Our method directly uses raw pixel values of the image as the matrix and models texture image denoising as a low rank matrix optimization problem. Besides the low rank assumption, we also utilize the assumption of continuity of natural images, by adding a total variation regularizer to the optimization objective function. Our results demonstrate that the TV regularizer indeed helps low rank texture denoising. Through extensive experiments, we have shown that our LRTD method works better than existing algorithms both in subjective visual terms and through objective quantitative measures.

Although our algorithm works robustly under a broad range of conditions and for a wide range of regular textures, it may fail if the conditions are too challenging or the assumptions are violated. Figure 6 shows some such cases. The denoising results in Fig. 6(c) and Fig. 6(e) are reasonable but not perfect. As mentioned earlier in Section 1, our algorithm is not designed to work on random textures. Although there has been work in the literature showing that it is possible to get a reasonable denoised result for



(a) Input 11.18 dB (b) Ref. [22] 23.47 dB (c) Ref. [23] 16.66 dB (d) Ref. [13] 22.24 dB (e) LRTD 30.01 dB

(f) Input 11.15 dB (g) Ref. [22] 20.70 dB (h) Ref. [23] 11.42 dB (i) Ref. [13] 21.06 dB (j) LRTD 26.28 dB

(k) Input 12.42 dB (l) Ref. [22] 22.45 dB (m) Ref. [23] 19.76 dB (n) Ref. [13] 25.31 dB (o) LRTD 29.12 dB

(p) Input 12.88 dB (q) Ref. [22] 23.22 dB (r) Ref. [23] 17.36 dB (s) Ref. [13] 23.90 dB (t) LRTD 26.72 dB

Fig. 4 Qualitative comparison of results of various methods (2). Left to right: input noisy image with $\sigma = 15, sr = 45\%$, denoising results of Ref. [22], KALS [23], ASALM [13], and our denoising result.

such examples as grass lawns, our algorithm is not designed to handle such cases. LRTD is effective for regular symmetric textures, but not for random textures which normally have high rank matrices.

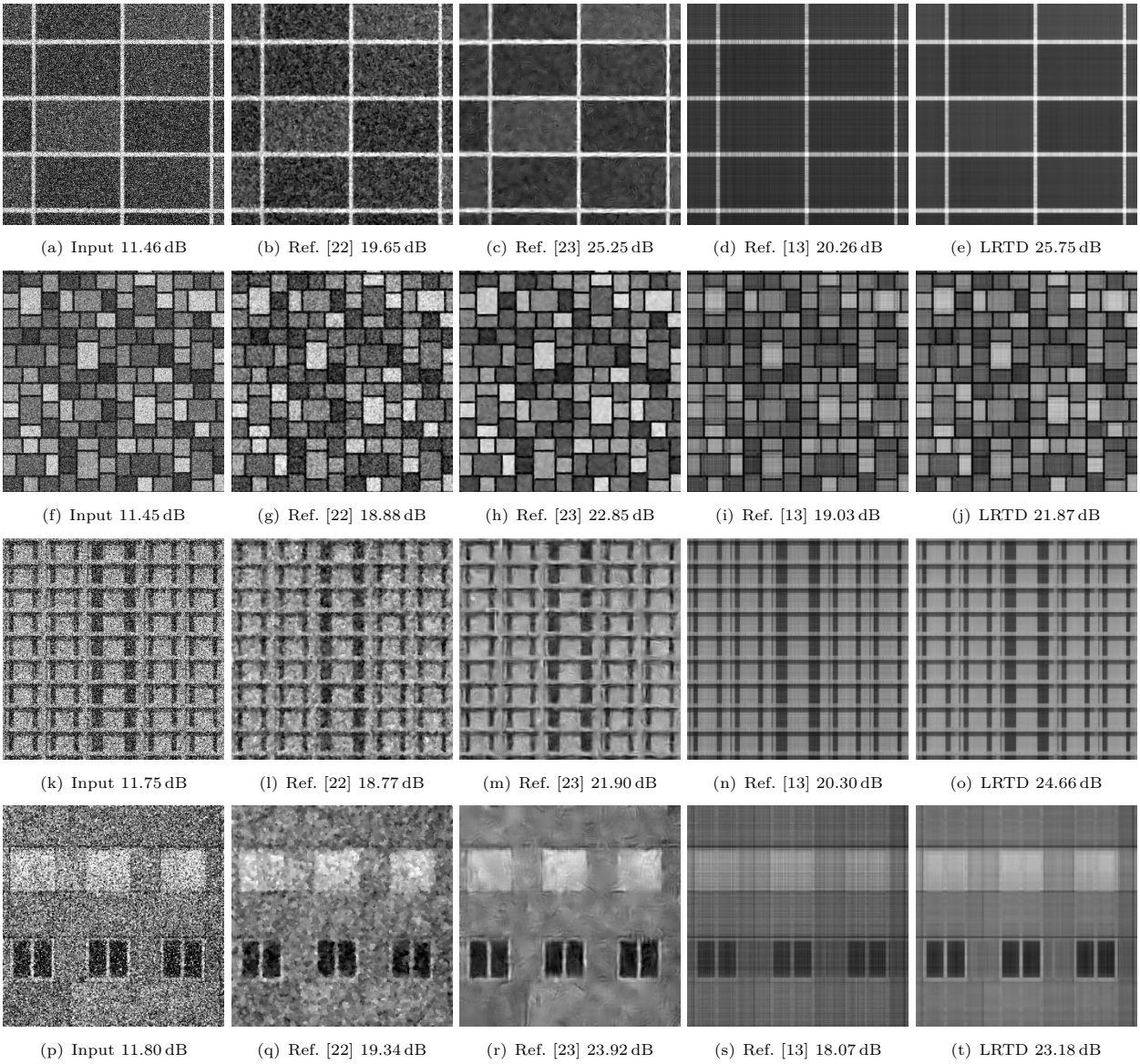
Clearly, a natural image of low rank texture may be deformed by the camera projection and undergoes a certain domain transformation (say affine or projective). The transformed texture, viewed as a matrix, in general is no longer low rank in the image domain. Nevertheless, by utilizing advanced convex optimization tools [10] from matrix rank minimization, we can recover a low rank texture from the deformed image and the associated deformation. As TILT (transform invariant low rank texture) [10] uses image interpolation during its iterations, simply running LRTD on TILT results would hurt image performance. Finding how to combine TILT with LRTD to enable a wider range of applications would be valuable future work.

Acknowledgements

We sincerely appreciate Zhouchen Lin and Xin Tong's help with valuable suggestions and comments for this paper.

References

[1] Rank, K.; Unbehauen, R. An adaptive recursive 2-D filter for removal of Gaussian noise in images. *IEEE*



(a) Input 11.46 dB (b) Ref. [22] 19.65 dB (c) Ref. [23] 25.25 dB (d) Ref. [13] 20.26 dB (e) LRTD 25.75 dB

(f) Input 11.45 dB (g) Ref. [22] 18.88 dB (h) Ref. [23] 22.85 dB (i) Ref. [13] 19.03 dB (j) LRTD 21.87 dB

(k) Input 11.75 dB (l) Ref. [22] 18.77 dB (m) Ref. [23] 21.90 dB (n) Ref. [13] 20.30 dB (o) LRTD 24.66 dB

(p) Input 11.80 dB (q) Ref. [22] 19.34 dB (r) Ref. [23] 23.92 dB (s) Ref. [13] 18.07 dB (t) LRTD 23.18 dB

Fig. 5 Qualitative comparison of results of various methods (3). Left to right: input noisy image with $\sigma = 60, sr = 10\%$, denoising results of Ref. [22], KALS [23], ASALM [13], and our denoising result.

(a) Input (b) 33.54 dB (c) Input (d) 27.67 dB (e) Input (f) 21.51 dB

Fig. 6 Examples with decreasing regularity and increasing randomness. (a), (c), and (e) input noisy images, with $\sigma = 10, sr = 10\%$; (b), (d), and (f) denoising results and their PSNRs.

Transactions on Image Processing Vol. 1, No. 3, 431–436, 1992.

[2] Chan, S. H.; Khoshabeh, R.; Gibson, K. B.; Gill, P. E.; Nguyen, T. Q. An augmented Lagrangian method for total variation video restoration. *IEEE Transactions on Image Processing* Vol. 20, No. 11, 3097–3111, 2011.

[3] Buades, A.; Coll, B.; Morel, J.-M. A non-local algorithm for image denoising. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 2, 60–65, 2005.

[4] Dabov, K.; Foi, A.; Katkovnik, V.; Egiazarian, K. Image denoising by sparse 3-D transform-domain

collaborative filtering. *IEEE Transactions on Image Processing* Vol. 16, No. 8, 2080–2095, 2007.

[5] Afonso, M. V.; Sanches, J. M. R. Blind inpainting using l_0 and total variation regularization. *IEEE Transactions on Image Processing* Vol. 24, No. 7, 2239–2253, 2015.

[6] Gastal, E. S. L.; Oliveira, M. M. Adaptive manifolds for real-time high-dimensional filtering. *ACM Transactions on Graphics* Vol. 31, No. 4, Article No. 33, 2012.

[7] Zhang, Q.; Xu, L.; Jia, J. 100+ times faster weighted median filter (WMF). In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2830–2837, 2014.

[8] Buades, A.; Coll, B.; Morel, J.-M. A review of image denoising algorithms, with a new one. *SIAM Journal on Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal* Vol. 4, No. 2, 490–530, 2005.

[9] Chatterjee, P.; Milanfar, P. Is denoising dead? *IEEE Transactions on Image Processing* Vol. 19, No. 4, 895–911, 2010.

[10] Zhang, Z.; Ganesh, A.; Liang, X.; Ma, Y. TILT: Transform-invariant low-rank textures. *International Journal of Computer Vision* Vol. 99, No. 1, 1–24, 2012.

[11] Dong, W.; Shi, G.; Li, X. Nonlocal image restoration with bilateral variance estimation: A low-rank approach. *IEEE Transactions on Image Processing* Vol. 22, No. 2, 700–711, 2013.

[12] Shabalin, A. A.; Nobel, A. B. Reconstruction of a low-rank matrix in the presence of Gaussian noise. *Journal of Multivariate Analysis* Vol. 118, No. 5, 67–76, 2013.

[13] Tao, M.; Yuan, X. Recovering low-rank and sparse components of matrices from incomplete and noisy observations. *SIAM Journal on Optimization* Vol. 21, No. 1, 57–81, 2011.

[14] Zhang, Y.; Liu, Y.; Li, X.; Zhang, C. Salt and pepper noise removal in surveillance video based on low-rank matrix recovery. *Computational Visual Media* Vol. 1, No. 1, 59–68, 2015.

[15] Zhou, Z.; Li, X.; Wright, J.; Candès, E. J.; Ma, Y. Stable principal component pursuit. In: Proceedings of IEEE International Symposium on Information Theory, 1518–1522, 2010.

[16] Liang, X.; Ren, X.; Zhang, Z.; Ma, Y. Texture repairing by unified low rank optimization. *Journal of Computer Science and Technology* Vol. 31, No. 3, 525–546, 2016.

[17] Candès, E. J.; Li, X.; Ma, Y.; Wright, J. Robust principal component analysis? *Journal of the ACM* Vol. 58, No. 3, Article No. 11, 2011.

[18] Wang, Z.; Zhang, J.; Chen, G. Mixture noise image denoising using reweighted low-rank matrix recovery. *Compuer Science* Vol. 43, No. 1, 298–301, 2016. (in Chinese)

[19] Lin, Z.; Ganesh, A.; Wright, J.; Wu, L.; Chen, M.; Ma, Y. Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix. Technical Report. University of Illinois at Urbana-Champaign, 2009.

[20] Candès, E. J.; Plan, Y. Matrix completion with noise. *Proceedings of the IEEE* Vol. 98, No. 6, 925–936, 2010.

[21] Djurović, I. BM3D filter in salt-and-pepper noise removal. *EURASIP Journal on Image and Video Processing* Vol. 2016, 13, 2016.

[22] Cai, J.-F.; Chan, R. H.; Nikolova, M. Fast two-phase image deblurring under impulse noise. *Journal of Mathematical Imaging and Vision* Vol. 36, 46–53, 2010.

[23] Wang, Y.; Szlam, A.; Lerman, G. Robust locally linear analysis with applications to image denoising and blind inpainting. *SIAM Journal on Imaging Sciences* Vol. 6, No. 1, 526–562, 2013.

Xiao Liang is currently a Ph.D. student in computer science and technology at the Institute for Advanced Study in Tsinghua University, Beijing, China. Her adviser is Prof. Harry Shum. She received her B.E. degree in electronic engineering from Tsinghua University. During her study, she interned at Microsoft Research Asia for over four years. Her research interests include texture processing, 3D computer vision and sparsity, and low rank matrix recovery.